

Introdução ao L^AT_EX

Seminário L^AT_EX - o Livro

Geraldo Xexéo¹ and Geraldo Xexéo^{1,2}

¹Departamento de Ciências da Computação

²Programa de Engenharia de Sistemas e Computação

Março 2020

Resumo

Esse texto é uma introdução ao \LaTeX escrita em português, criada como resultado de um seminário de introdução ao \LaTeX realizado na época do Covid-19.

Capítulo 1

A Cadeia de Processamento de Texto

Os computadores foram criados para fazer contas, mas na verdade eles manipulam apenas símbolos. Rapidamente seus usuários perceberam que podiam ser usados para manipular textos, como código, e formatá-los de alguma forma para impressão. O processamento de texto é possivelmente a área de software com o maior número de usuários, já que todo usuário de computador, alguma vez, escreveu algo e enviou para alguém, nem que seja um simples e-mail. Para isso, foram criados vários tipos de programas, que cumprem funções como as descritas na figura 1.1. Basicamente, o que a figura mostra é um arquivo de texto pode ser editado, visualizado, processado para impressão ou processado de formas adicionais. Cada sistema de processamento de texto faz, prioritariamente uma dessas funções.

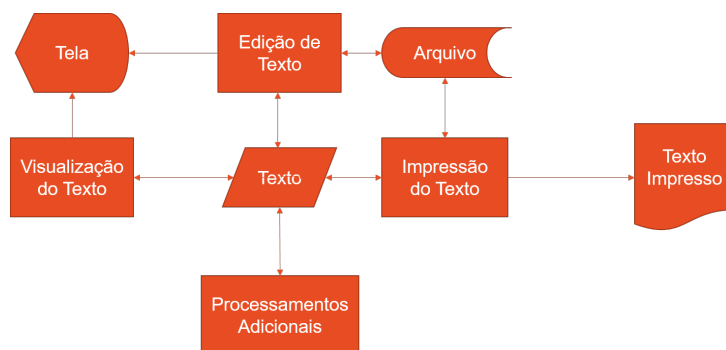


Figura 1.1: A cadeia de processamento de texto

1.1 Tipos de Sistema de Processamento de Texto

1.1.1 Editores de Texto

Editores de texto são programas que permitem ao usuário editar arquivos que são de **texto puro**. Texto puro é um conceito que mudou. Inicialmente significava que havia um mapeamento um para um entre o que você encontrava no arquivo, uma sequência de caracteres codificados em ASCII¹. Em ASCII, por exemplo, a letra “A” é representada pelos bits “1000001” e a “a” por “1100001”. Atualmente são usadas codificações que permitem que um caracter ou símbolo seja representado por uma sequência mais longa de bits, por meio de *escape codes*, por exemplo UTF-8², o que significa que os editores de texto, normalmente, não representam mais perfeitamente o arquivo em disco. Os sistemas de codificação atuais são normalmente extensões do ASCII, isto é, os 128 códigos de 1 byte do ASCII original ainda são válidos. A figura 1.2 mostra a função de um editor de texto na cadeia de processamento de texto.

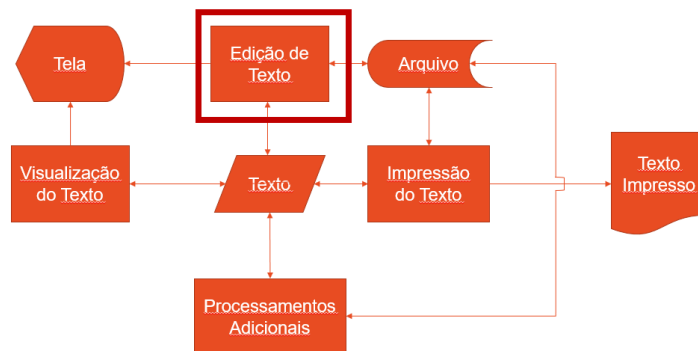


Figura 1.2: Função de um editor de texto na cadeia de processamento de texto.

Editores de texto foram necessários assim que trocamos as entradas por cartão e fita, que eram editados em máquinas não conectadas ao computador, por terminais ligados diretamente aos mesmos. Os primeiros editavam linha a linha, a seguir outros exigiam que o usuário gerenciasse o *buffer*, o seja, a

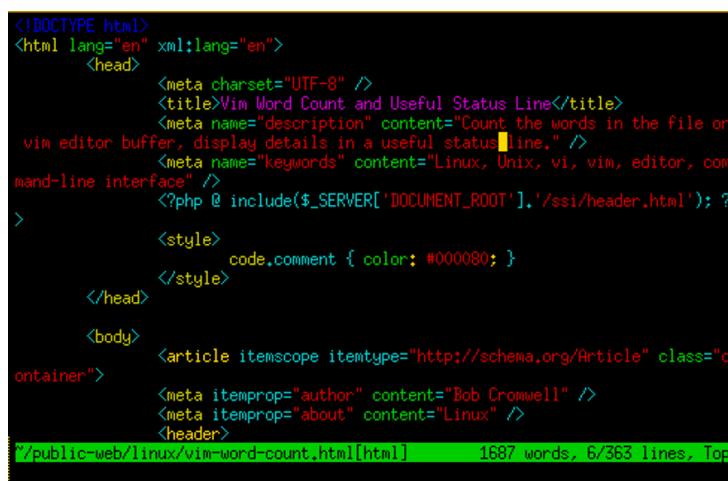
¹ASCII é um padrão que associa uma letra, e outros símbolos usados em arquivos, a um byte. Seu nome significa American Standard Code for Information Interchange. Baseado no alfabeto inglês, originalmente usava apenas 128 símbolos (7 bits), não possuindo os caracteres acentuados de outras línguas.

²Unicode Transformation Format, que permite codificar 1.112.064 símbolos

parte do arquivo que estava em memória. Com o tempo chegamos a versões semelhantes as atuais.

Atualmente editores de texto tem um conjunto complexo de funções de busca, substituição, etc.

Exemplos de editores de texto atuais são o Notepad, que vem por *default* com o Windows, o *vi* e o *vim*, Notepad++, EditPlus, TextEdit e o poderosíssimo Emacs. A figura 1.3 mostra um exemplo to *vim*.



```
<!DOCTYPE html>
<html lang="en" xml:lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Vim Word Count and Useful Status Line</title>
    <meta name="description" content="Count the words in the file or
vim editor buffer, display details in a useful status line." />
    <meta name="keywords" content="Linux, Unix, vi, vim, editor, com
mand-line interface" />
    <?php @ include($_SERVER['DOCUMENT_ROOT'].'/ssi/header.html'); ?
  >
  <style>
    code,comment { color: #000080; }
  </style>
</head>
<body>
  <article itemscope itemtype="http://schema.org/Article" class="c
ontainer">
    <meta itemprop="author" content="Bob Cronwell" />
    <meta itemprop="about" content="Linux" />
  <header>
    /public-web/linux/vim-word-count.html[html] 1687 words, 6/363 lines, Top
```

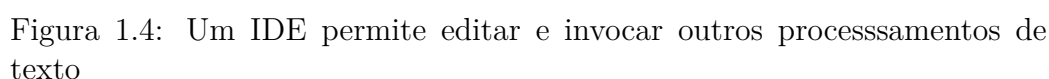
Figura 1.3: O editor de texto vim

1.1.2 IDEs

Uma IDE, ou “Integrated Development Environment”, é uma extensão lógica da ideia de editor de texto criada originalmente para programadores, fornecendo serviços adicionais a edição de texto, como compilação, *debugging*, controle de versão, normalmente por meio de interfaces com os programas que fazem isso.

Uma IDE cobre a parte de edição e processamento de texto da cadeia de processamento de texto, como visto na figura 1.4.

Exemplos de IDE são o Atom, o Visual Studio, o T_EXStudio e o próprio Emacs. Com a evolução dos editores de texto, a fronteira entre editores e IDEs ficou indefinida, muitas vezes dependendo do uso que o usuário faz do programa. A figura 1.5 mostra um exemplo do T_EXStudio.



Já que era possível editar textos, porque não imprimi-los de forma adequada? Essa ideia levou a criação de programas de tipografia, que faziam a tradução de um arquivo texto, com marcações adequadas, para um outro arquivo que fosse interpretado em uma impressora (ou outras máquinas mais sofisticadas de *typesetting* digital).

Essas marcações adequadas são como comandos, e um arquivo de texto marcado se assemelha a um programa de computador, por possuir palavras código que dão os comandos necessários. Sistemas desse tipo não possuem

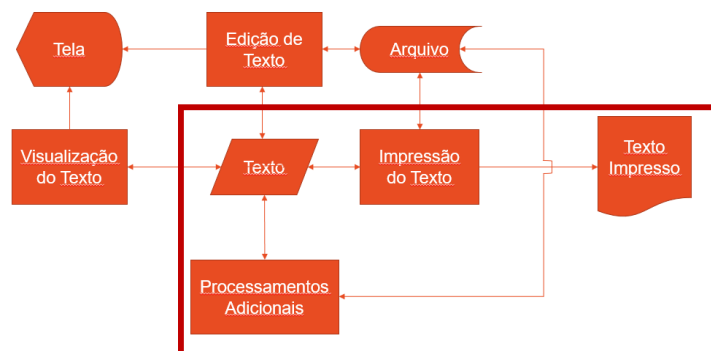


Figura 1.6: Sistemas de tipografia imprimem e fazem processamentos adicionais.

editores associados e são normalmente ativados por linha de comando.

Exemplos de sistemas de tipografia são o troff, o $\text{T}_{\text{E}}\text{X}$ e o $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$. A figura 1.7 mostra o $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ sendo usado em linha de comando.

```

Command Prompt
F:\Github\Seminario-LaTeX>latexmk
RC files read:
  .latexmkrc
Latexmk: This is Latexmk, John Collins, 17 Apr. 2020, version: 4.69a.
Latexmk: applying rule 'pdflatex'...
Rule 'pdflatex': The following rules & subrules became out-of-date:
  'pdflatex'
-----
Run number 1 of rule 'pdflatex'
-----
Running 'pdflatex -recorder "artigocomabstract.tex"'
-----
This is pdfTeX, Version 3.14159265-2.6-1.40.21 (MiKTeX 2.9.7400 64-bit)
entering extended mode
(artigocomabstract.tex
LaTeX2e <2020-02-02> patch level 5
L3 programming layer <2020-04-06>
("F:\Program Files\MiKTeX 2.9\tex\latex\base\article.cls"
Document Class: article 2019/12/20 v1.41 Standard LaTeX document class
("F:\Program Files\MiKTeX 2.9\tex\latex\base\size10.clo"))
("F:\Program Files\MiKTeX 2.9\tex\latex\base\fontenc.sty")
("F:\Program Files\MiKTeX 2.9\tex\generic\babel\babel.sty"
("F:\Program Files\MiKTeX 2.9\tex\generic\babel\babel.def"
("F:\Program Files\MiKTeX 2.9\tex\generic\babel\txtbabel.def"))
*****
* Local config file bblopts.cfg used
*
("F:\Program Files\MiKTeX 2.9\tex\latex\arabi\bblopts.cfg")
("F:\Program Files\MiKTeX 2.9\tex\latex\babel-english\english.ldf")

```

Figura 1.7: O $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ sendo usado

1.1.4 Processadores de Texto

Principalmente com o advento do micro-computador, ficou claro que uma das principais utilidades do computador seria permitir a criação de textos a

serem publicados, logo um editor de texto deveria ser estendido para suportar funções como colocar palavras em negrito, itálico, selecionar fontes, etc.

Processadores de texto cumprem grande parte das funções na cadeia de processamento de texto, como mostra a figura 1.8.

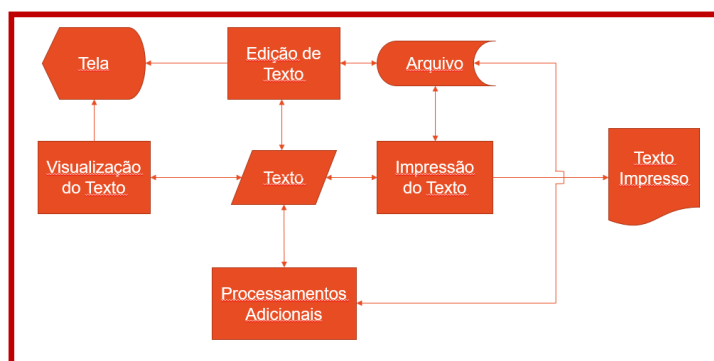


Figura 1.8: Os processadores de texto na cadeia de processamento de texto.

Com a evolução dos terminais de computadores, micro-computadores e placas de vídeo e monitores, os processadores de texto evoluíram de sistemas que mostravam alguma coisa do que estava sendo prevista para o texto, como caracteres em bold, para sistemas que permitiam uma visualização prévia, até chegar a edição pelo conceito de WYSIWYG, lido “uiziwig”, que mostra na tela quase que exatamente o que será visto na edição final, sendo as diferenças mínimas e quase imperceptíveis causadas por questões tecnológicas e da diferença entre papel e tintas e monitores.

Atualmente o Word é o processador de texto que domina amplamente o mercado, e seus principais concorrentes são sistemas de código aberto, como o LibreOffice Writer ou o Apache OpenOffice Writer. Um tela do Word é mostrada na figura 1.9.

1.1.5 Sistemas de Autoria

Sistemas de autoria são uma evolução interessante dos processadores de texto, ou dos IDEs, voltadas para autores de livros, roteiros, etc. que não só permitem editar o texto, em formatos específicos, como também guardar informações como fichas de personagens, descrições de cena, *storyboards*, etc... Eles podem cobrir toda a cadeia de processamento de texto.

Exemplo de sistemas de autoria são o Scrivener, apresentado na figura 1.10, o Final Draft e o Celtx.

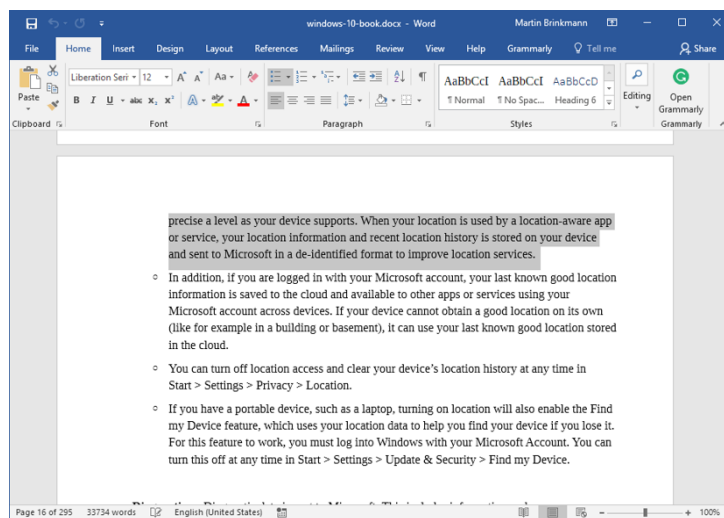


Figura 1.9: O Word é o principal processador de texto do mercado.

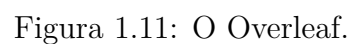
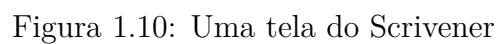
1.1.6 Sistemas Colaborativos de Edição

Esses sistemas aparecem cedo, porém se expandem principalmente com o fortalecimento da internet. Eles permitem que mais de uma pessoa edite um arquivo ao mesmo tempo. Atualmente o mais conhecido é o Google Docs, que é um processador de texto limitado em funcionalidade mas muito fácil de usar.

O ShareLateX foi um sistema colaborativo de edição voltado para o \LaTeX que acabou sendo responsável por um renascimento do uso do \LaTeX na academia. Acabou sendo incorporado ao concorrent Overleaf, que aparece na figura 1.11.

1.1.7 O que é o \LaTeX

\LaTeX é um sistema de typesetting baseado no \TeX apoiado com outros programas, como o biber que permite usar arquivos de texto marcados para criar arquivos a serem impressos, ou no formato PDF, seguindo regras de composição (*typesetting*) e usando fontes detalhadamente criadas para reproduzir a qualidade de fontes utilizada na composição manual, incluindo especialmente as ligaduras, que são caracteres especiais que representam dois ou mais caracteres de forma visualmente mais elegantes, como mostradas na figura 1.12.



<i>fi</i>	→	<i>fl</i>	<i>AE</i>	→
<i>ff</i>	→	<i>ff</i>	<i>ae</i>	→
<i>ffi</i>	→	<i>ffi</i>	<i>OE</i>	→
<i>fl</i>	→	<i>fl</i>	<i>oe</i>	→
<i>ij</i>	→	<i>ÿ</i>	<i>LATEX</i>	→

Figura 1.12: Exemplos de ligaturas de fontes do T_EX.